2vs2 Multi-Agent Reinforcement Learning in Rocket League

AASMA - 2nd Semester - 2022

Instituto Superior Técnico

Carlos Sequeira
87638
MECD
carlos.r.sequeira@tecnico.ulisboa.pt

Filipa Costa
92626
MMA
filipabcosta@tecnico.ulisboa.pt

ABSTRACT

When compared to football, Rocket League introduces special mechanics, not just for the usage of vehicles to kick a ball, but also for the use of a rocket boost on these cars, which adds a third dimension to a normal football-like environment.

This paper outlines the concepts for implementing a multi-agent Reinforcement Learning system which ultimately should result in team behaviour while creating scoring oportunities against a pre-defined opponent.

We begin with a quick overview and problem definition in Section 1. In section 2, we define the multi-agent system. In section 3, we provide the evaluation procedure. The results and discussion are shown In section 4. Section 5 and 6 describe the conclusion and future work.

We observed that the best performing agent was the one that has more experience and the agent always improves with more training. Furthermore, we discovered that the team formed of the most experienced agents wins against the one composed of the Psyonix robots from the Rocket League game, despite now showing much team Behavior.

KEYWORDS

Rocket League, Multi-Agent System, Reinforcement Learning, Team Behavior.

1. Introduction

Rocket League is a sport-based video game developed by Psyonix in which players control a rocket-boosted car and attempt to hit a ball into the opponents' net while preventing the adversaries from doing the same in their own net.

Our project is defined as a long horizon problem in which a team of two agents need to decide on the best sequence of decisions that lead to a maximization of the objective. For this reason, Reinforcement Learning is applied so that agents can learn through experimentation while shaping a policy to attempt the maximization of the reward for each episode, in the minimum amount of time.

1.2 Problem definition

Our multi-agent system is composed of two agents that cooperate with each other to achieve a goal against a team composed by the same two agents. The agents begin with no knowledge about which moves are good, so they must learn by trial and error, while taking into account the reward provided by the environment.

Our objective is to implement a multi-agent system, where each agent is not only able to learn how to cooperate with each other, but also to determine how to score and how to score fast.

2. Multi-agent System

2.1. Environment

The environment is a normal Rocket League closed-up field with two agents, one Rocket League bot, one ball, 34 boost pads divided into twenty-eight small boost pads (that give 12 boost) and six big boost pads (that give 100 boost), and 2 goals. Both ball and players cannot leave the pitch, allowing for continuous gameplay and simplified end-to-end learning.

Figure 1: Rocket League Gameplay in a 2vs2 bot game.

2.2 Initial State

At the start of each episode, all the four players and ball are placed on the field in a random game situation. Every time one episode ends, the game's state is reset to another random game state. This allows players to learn how to cooperate with one another regardless of the game state and avoids overfitting.

2.3 Terminal Condition

The terminal conditions determine when an episode ends, which happens if any of the following defined conditions is true:

- A **goal was scored** by any team;
- **30 seconds** have elapsed without touching the ball;

It is important to reinforce that if the ball is touched, then the game can run until a goal is scored. This is done in order for the agent to understand by its own that scoring is better than touching the ball.

2.4 Actuators

The agent's actuators are defined as the possible actions than an agent can perform at any given moment. The available actions for each agent are all discrete and are defined as a combination of the following individual components:

- **Throttle** ∈ {-1,0,1} (-1 is backwards, 1 is forwards);
- **Steer** (or **Yaw**) ∈ {-1,0,1} (rotation around z axis, -1 anti-clockwise, 1 clockwise);
- **Boost** ∈ {0,1}

2.5 Sensors

The agent's sensors are defined as the observations that each agent perceives of the state at any given moment. In this case, the game state is totally perceived, and the observations are divided along these lines:

- Players Information:
  - Position / relative Forward Direction / relative Up Direction / Linear Velocity / Angular Velocity / Boost amount / Is on the ground / Has_Flip (second jump) / Is teammate, opponent or Self / Ball_Touched / Is_Demoed
- Ball Information:
  - Position / Direction / Linear Velocity / Angular Velocity
- Boosts Information:
  - Position / Boost Amount / Is_Available

Note that the goal positions are not present on the observation builder as the reward functions will take care to teach the agents where they can score the goals at.

2.6 Rewards

The agent's objectives will be determined by the defined rewards that will shape each agent's policies. The rewards are defined as follows:

- Ball touched - returns +0.2;
- Velocity to Goal - returns $e^{(-||x_{goal}-x_{ball}||)/v_{max}}$ where x_{goal} and x_{ball} are the positions of the goal and ball, and v_{max} is the maximum possible ball velocity;
- Goal scored - returns +10;

There is also a reward setting called team spirit that defines the proportion of individual rewards that are distributed across the team instead of just the player. When this setting is equal to 0, the rewards are not splitted across the team, when this setting is equal to 1, all the rewards are distributed across the team. During training this setting was initially set to 0.3 and then progressively changed to 1 between 300 million steps and 400 million steps of training, after we observed that the agent started to touch the ball consistently.

2.7 Policy Algorithm

To solve our problem a default implementation of Proximal Policy Optimization (PPO) [2] will be used as our learning model. The implementation of this model was based on a similar project of Reinforcement Learning applied in Rocket League [4].

3. Evaluation

For the evaluation step we decided to perform a tournament between different versions of the agent and the best in-game hardcoded bot developed by Rocket League.

After training for 900 million steps, corresponding to around 16667 hours of in-game time, we got the following average of episode length for the corresponding number of training steps present on Figure 2.

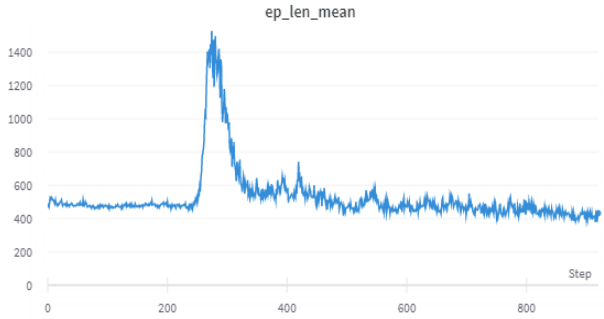

Figure 2: Episodes lengths for steps of training

By analyzing the graph we can reach three conclusions:

1. It took about 250 million steps for the agent to start touching the ball consistently.
2. It took about 280 million steps for the agent to start scoring consistently.
3. After 340 million steps the agents progressed slowly and learned to score faster and faster.

That said, we decided to select 5 different versions of the agent: (1) The first saved model with 10 million steps of training, which will behave approximately to a random agent and will serve as a baseline; (2) Agent with 280 million steps, representing the peak of the episode length in training, which reflects an agent which understands how to touch the ball but not so much how to score goals; (3) Agent with 450 million steps which stands right in the middle of the full training length; (4) Agent with 800 million steps to verify how close this version is to the final version; (5) Agent with 900 million steps, which we call the Latest version representing the agent with the most training. We also selected for the tournament an all-star Rocket League bot as the second baseline.

An interesting aspect will be to verify if the agent can defeat this in-game bot with a much reduced action space, which enables the Psyonix bot to fly and make air plays and which was not trained by our agent.

Once the players were selected, we needed to decide the tournament format. Unfortunately, the used frameworks do not offer the possibility to play with multiple instances or to speed up the game as it would lead to loss of performance. For these reasons we decided to implement a format proposed by D. Budden [3] with a slight variation, which consists of a round-robin tournament where:

1. Each player plays against all the others
2. Based on the previous results, a BO3 is performed between 3rd and 2nd place to see which team advances to the final with the 1st place
3. A BO3 to decide the winner of the tournament

This method was found to be robust when the objective is to find out the best agent for a small sample of games.

4. Results and Discussion

We will start with evaluating the team performance of each agent, by looking at the matches results and using simple metrics such as goals scored, goals conceded, goals difference, victories and defeats.

We will also look at the average percentage of time spent behind the ball for each agent, as well as the average time spent in Supersonic Speed, Boost Speed, and Slow Speed for each agent. Better agents should spend more time behind the ball, so they can both attack and defend, and also be as fast as possible while moving though the field.

Then, in order to evaluate team behavior, we will look at the average distance each agent has to its teammates and the ball, as well as the average number of goals that were assisted.

Finally, we will verify the agent that won the tournament.

4.1. Tournament Performance

Looking at match results is a simple way to evaluate a team's performance.

Table 1 shows that total scored goals rise with agent experience while total conceded goals fall (with the exception of agent "800", which may be attributable to the low number of games played).

Because it still lacks training, the agent "First" did not score any goals, behaving similarly to a random agent.

As the agents gain more experience, they appear to recognize that scoring a goal is the most rewarding, while conceding a goal is the least rewarding, as they begin to score more goals and defend against conceding a goal. This comes with the developed notion of where both goals are and how to attack/defend them.

The team comprised by the agents "Latest" appears to be behaving the best, as the goal difference is the

highest. It is also clear that the goal difference is greater than the Psyonix agent, meaning that the “Latest” agent has learned a good understanding of the game. This can also be observed by the created vídeo.

Agent	TSG	TCG	GD	V	D
First	0	80	-80	0	5
280	10	38	-28	1	4
450	35	14	21	2	3
800	36	16	20	4	1
Latest	48	12	36	4	1
Psyonix	49	18	31	4	1

Table 1: TSG- Total Scored Goals, TCG- Total Conceded Goals, GD- Goals Difference, V- Victories, D- Defeats

In Figure 3, we graphed the difference in goals suffered by each agent versus the other team of agents to dig deeper into the differences in goals performed by each agent.

Each line represents one of the five different teams of agents we trained, the x-axis represents the team of agents who played the game against them, and the y-axis shows the difference in goals scored. For example, when the team composed by 2 agents “Latest” (purple line) plays against the one composed by 2 agents “First” (Agent Against 1), the goal difference is higher than +20, meaning that the purple team won the game.

It is also worth mentioning that when the same team of agents competes against themselves, the goal difference is set to zero.

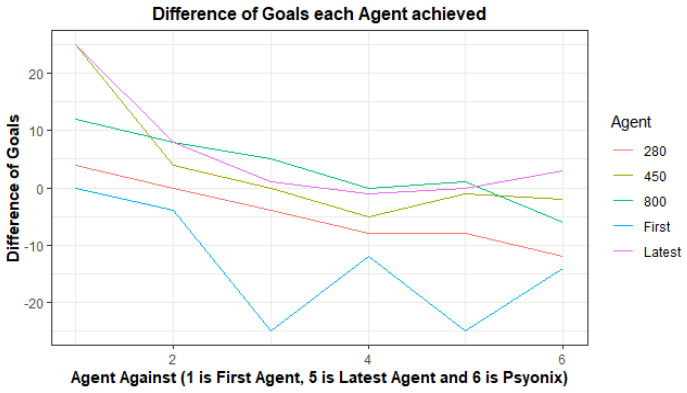

Figure 3: Difference of goals each agent achieved.

According to Figure 3, Agent "First" is the worst, followed by agent "280", as they never score more than they concede. The only exception is the victory of agent "280" against "First" by four goals.

There are some games that differ between agents "450", "800", and "Latest". For example, agent "450" always has a lower goal difference than agent “Latest”, and both agents "450" and "800" perform worse against Psyonix than the "Latest" agent when playing versus Psyonix.

However, agent "800" is better than agent "Latest" when against agent "450", and in the game of agent "800" versus. agent "Latest," the agent "800" won.

Agents “800” and “Latest” seem to have a similar behavior, which shows the slow progress in the later stages of training.

Table 2 shows the percentage of time each player spends, on average, behind the ball.

This percentage must be assessed since the player must be behind the ball in order to kick the ball into the opponent's goal and in order to defend its own goal.

Indeed, it is noticeable that the proportion behind the ball increases with the agent's experience, indicating that the agent is learning this information.

Agent	Percentage Behind Ball
First	31,00%
280	44,86%
450	77,17%
800	79,84%
Latest	79,90%
Psyonix	75,98%

Table 2: Average behind ball percentage per player for each agent.

Surprisingly, we can also find in Table 2 that the agent "Latest" is more times behind the ball than the agent Psyonix, which may be a motive why "Latest" can defeat Psyonix even without air plays.

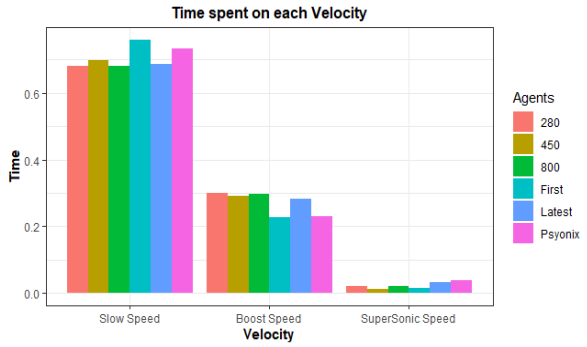

Figure 4: Percentage of time spent by each agent on each type of velocity.

Figure 4 shows that as the agents gain experience, they become faster.

The ones who spend the most time at SuperSonic Speed (the fastest speed available) are Psyonix and “Latest”.

On the other hand, the "Latest" agent spends more time in Boost Speed (medium speed) than the Psyonix, and the Psyonix spends more time in Slow Speed, implying that, overall, the "Latest" agent appears to be quicker. The slowest agent is the “First”, what is expected with low experience.

There also doesn't appear to be much of a difference between the agents "280", "450", "800", and "Latest," implying that having higher speed is easy to learn.

4.2. Team Behavior

Assessing team behavior is difficult because it requires identifying how much of a team's performance is due to team performance and how much is due to the players' individual abilities.

Observing replays and manually judging team behavior is a simple technique to evaluate it, but it is also time consuming and subjective [6].

In this project, we analyzed team behavior by extracting some metrics from replays. Assisted goal percentage [5], average distance each agent has to its teammates, and average distance each agent has to the ball are all promising indicators for team behavior in Rocket League.

When looking at the assisted goal percentages for each agent in Figure 5, it is clear that as the agent gains experience, he recognizes that assisting goals rather than scoring on his own is better for the team. This can be concluded since from Table 1, it is possible to see that the agents who have a higher percentage of goals assisted are also the ones who have a higher number of total scored goals. Although the value is still lower than the Psyonix agent, standing just below an assist for each four goals scored. This implies that our learning system does not suffice for the agents to learn how to play together by themselves, at least without a tremendous amount of training.

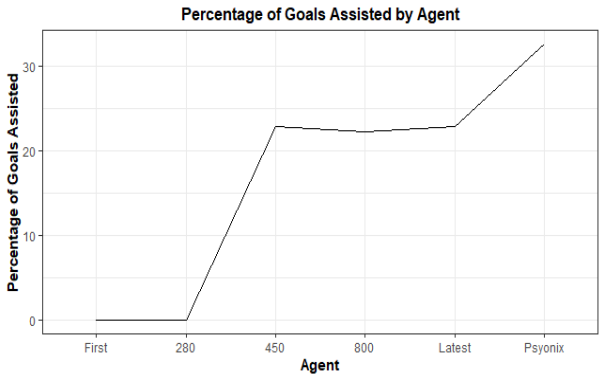

Figure 5: Percentage of goals assisted by player, for each agent.

Another important measure is the average distance each agent has to the ball and to its teammate.

Figure 6 shows that the average distance between each agent and the ball drops for the first two agents, but remains relatively constant for the other three. This

can be explained by the fact that as the agent gains experience, he will be able to locate the ball and maintain a reasonable distance.

On the other hand, all agents appear to have the same distance to the team mate. This can be a problem because the distance between teammates should rise as the agent gains expertise, allowing the team to cover more ground of the field and develop a better strategy.

In Figure 6 it is also noticeable that with the exception of the Psyonix agent, the distance between the agents and the ball is always greater than the distance between the agents and its teammate. This can be an issue because it implies both players want to catch the ball rather than wait for the other player to pass the ball and score, rendering the approach inefficient and preventing teamwork.

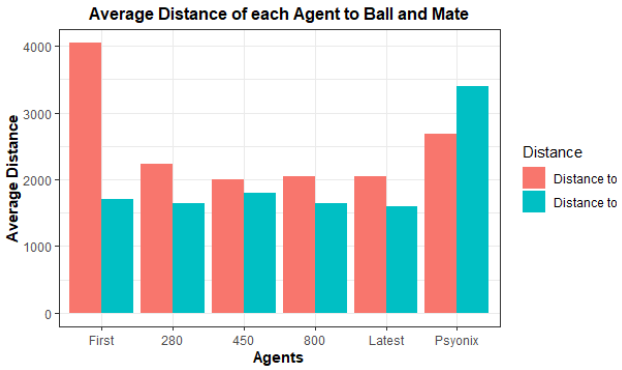

Figure 6: Average Distance each agent has to ball and to its teammate.

Finally, as a side note, this indicates that the players are not functioning as a team, because despite the increase in proportion of goals assisted, the percentage of goals assisted is lower when compared to Psyonix agent, and the distance to the ball is always greater than the distance to mate.

4.3. Tournament Results

Finally, the results derived by the second and third point described in the section 3.Evaluation of the round-robin tournament, are shown in Table 3 in order to determine the winner of the tournament.

	Game 1	Game 2	Game 3	Winner
First Round				
Psyonix vs 800	3:4	3:1	1:8	**800**
Second Round				
800 vs Latest	3:4	5:3	6:7	**Latest**

Table 3: BO3 between 3rd and 2nd place to see each team advances to the final with the 1st place (First Round) and to decide the winner of the tournament (Second Round).

Final classification:

1. “Latest” Agent
2. “800” Agent
3. Psyonix
4. “450” Agent
5. “280” Agent
6. “First” Agent

5. Conclusion

The developed agent successfully managed to defeat the baselines, both the random agent, which was simulated by the “First” agent. It also defeated the in-game best available bot, which had the possibility to perform air plays that were not feasible to our agent due to the reduced action space. For this reason, this win proved to be a big accomplishment, and showed that the agent was able to generalize to unseen conditions and situations.

Also, it is noticeable that there was a continuous improvement of the performance of the agent. Unfortunately, in terms of team behavior, our agent only managed to progress up to a certain point, which actually interesting highlights that were provided in the developed video, but it still shows that there are too many double commits, meaning that both members of the team are trying to perform the same task and cover the same ground of the field.

In general the results were very promising, as a simple Reinforcement Learning system was able to achieve

interesting results, even with a reduced amount of possible actions when compared to the Psyonix agent. To further improve team behavior we need to change this system and make it more complex.

6. Possible Future Work

Provide rewards for team behavior, specially rewards that incentivise division of labor and more assists.

Improve state setters to achieve unusual game states.

Try different learning systems and compare results. Mainly, teams composed of two different agents with different personalities that complement each other.

Give all possible actions to the agent and test the system with aerial plays, especially due to the jump action.

REFERENCES

[1] Siqi Liu, Guy Lever, Zhe Wang, Josh Merel, S. M. Ali Eslami, Daniel Hennes, Wojciech M. Czarnecki, Yuval Tassa, Shayegan Omidshafiei, Abbas Abdolmaleki, Noah Y. Siegel, Leonard Hasenclever, Luke Marris, Saran Tunyasuvunakool, H. Francis Song, Markus Wulfmeier, Paul Muller, Tuomas Haarnoja, Brendan D. Tracey, Karl Tuyls, Thore Graepel, and Nicolas Heess. From motor control to team play in simulated humanoid football. 5 2021.

[2] John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms, 2017.

[3] D. Budden, P. Wang, O. Obst, and M. Prokopenko, “Simulation leagues: Analysis of competition formats,” in Lecture Notes in Artificial Intelligence (Subseries of Lecture Notes in Computer Science), vol. 8992. Springer, Cham, 2015, pp. 183–194. [Online]. Available: http://link.springer.com/10.1007/978-3-319-18615-3_15

[4] R. Avril, “Necto” https://github.com/Rolv-Arild/Necto

[5] Yannick Verhoeven, and Mike Preuss. On the Potential of Rocket League for Driving Team AI Development, 2020

[6] R. Rein and D. Memmert, “Big data and tactical analysis in elite soccer: future challenges and opportunities for sports science,” p. 1410, dec 2016. [Online]. Available:http://springerplus.springeropen.com/articles/10.1186/s40064-016-3108-2